A LINEAR PROGRAMMING APPROACH TO WEAPON ALLOCATION

by

MAJOR WILLIAM T. HODSON III
MAJOR WILLIAM G. GOODYEAR
CAPTAIN WOLFHART B. GOETHERT

RESEARCH REPORT 71—1
MARCH 1971

UNITED STATES AIR FORCE ACADEMY
COLORADO 80840

52

A LINEAR PROGRAMMING APPROACH TO WEAPON ALLOCATION

by

Major William T. Hodson III
Major William G. Goodyear
Captain Wolfhart B. Goethert

UNITED STATES AIR FORCE ACADEMY

RESEARCH REPORT 71-1

MARCH 1971

Editor: Major Donald E. Sheehan, USAF

The research reported in this document was made possible in part by support extended to the faculty of the United States Air Force Academy by Headquarters Strategic Air Command.

Additional copies of this document may be obtained by writing to the Director of Faculty Research, United States Air Force Academy, Colorado 80840.

PREFACE

The models discussed in this paper were developed for the Future Force Structure Studies and Analysis Branch, DCS/Plans, Hq Strategic Air Command during the period from September 1969 through July 1970.

Although each of the authors was involved in the total project, Major Goodyear was primarily responsible for the concept formulation and application, Major Hodson for the development of the mathematical models, and Captain Goethert for the computer implementation.

While the models were developed for a strategic scenario, there would seem to be no reason why they could not be applied with equal success in the tactical arena. In fact, they are rather general resource allocation models, and need not be tied to military applications at all. The reader interested in applying this work to other allocation problems should replace the word "weapon" with "resource" and "target" with "task," whenever he encounters them in the text.

The authors would like to express their thanks to the Future Force Structure Branch for their continuing support in this project.

TABLE OF CONTENTS

Abstract vi

I. Introduction 1

The Weapon Allocation Problem 1

Importance of Good Allocation Methods 1

Comparing Arsenals 1

Detailed Allocations 2

Military Target Allocation 2

Strategic Retaliatory Scenario 2

Input Data 3

Targeting Philosophy 4

The Output 5

Sequenced Allocations 5

Allocation Within Categories 5

Variations 6

II. The Linear Programming Models 7

Mathematical Preliminaries 7

Allocations 7

Linear Programming 8

Computing Probability of Survival 8

An Approximation 9

The Linear Programming Models for Allocation Among Categories 11

The Basic LP Model 13

Interpretation of Results 14

The Advanced LP Model 14

The Alternate Advanced LP Model 17

The "Sufficient Weapons Condition" LP Model 18

Other Constraints 20

The Integer Programming Model for Allocation Within a Category 20
Notation 21
The Allocations 21
The Integer Programming Problem 22
Further Refinements 23
III. The Computer Model 25
General Description 25
Problem Size 25
Running Time 25
Attack Sequence 25
Program Input 26
Computer System 27
Program Restrictions 28
Detailed Description 28
Input Processing 28
Sort Procedure 30
Generation of Arrays A and B 31
Objective Function 33
Linear Programming Algorithm 33
Inversion Method 34
Output 36
General Overview 37
Future Configurations 38
Appendix 41
List of References 45

ABSTRACT

A two-step optimization procedure using linear programming is employed to obtain a near-optimal solution to the large-scale, multiple-weapon type allocation problem.

Initially, the target system is partitioned into target categories, each of which contains targets of equal worth and similar characteristics. Then, depending upon the requirements of the particular problem addressed, one of three different linear programming models is used to allocate the available supply of weapons among the target categories. Instead of inputing a point value for each target category, as is frequently done in other allocation models, in these models the user indicates a desired ratio of the probability of survival of the various categories.

After the allocation of weapons among the categories has been accomplished, an integer programming model is used to assign weapons to individual targets within each category so as to minimize the average probability of survival of the category.

These models have been programmed for an IBM 7090 computer at Headquarters Strategic Air Command.

CHAPTER I

INTRODUCTION

THE WEAPON ALLOCATION PROBLEM

The general problem of weapon allocation is that of assigning weapons to targets in such a way as to achieve a desired level of destruction.

The ideal situation would occur, of course, if for each target a single weapon of precisely the right yield were available and the weapons were assured of impacting on their targets. Since, in reality, this case never occurs, the problem becomes one of deciding either how to minimize the number of weapons required to inflict a given amount of damage, or how to maximize the amount of damage inflicted with a fixed supply of weapons.

The problem of minimizing the expenditure of weapons frequently occurs in sub-optimization exercises, and also in cases in which a sufficient number of weapons are available to achieve the desired level of destruction to all targets in a target system. However, the problem which faces the strategic planner more frequently in considering possible future force structures is that of maximizing damage to an entire target system with various arsenals of weapons, after having suffered a loss of weapons through a prior enemy strike.

IMPORTANCE OF GOOD ALLOCATION METHODS

Comparing Arsenals

When different combinations of types and numbers of weapons are applied to the same target system with the same targeting philosophy, it is possible to measure the effect of adding, retaining, or deleting certain weapon systems. Such a measurement, however, is meaningful only if an optimal (or very nearly optimal) allocation method is used, since an inefficient procedure may easily lead to erroneous conclusions in comparing two hypothetical arsenals.

Detailed Allocations

In preparing an actual detailed assignment of weapons to targets, as opposed to an aggregate allocation (as is done in this paper, for example), there may be operational or subjective reasons for the force applicator to deviate from the aggregate analysis. However, only if he has an algorithm which gives him a very good indication of what the optimal aggregate allocation is, will he be able to make a meaningful evaluation of the quality of his plan.

MILITARY TARGET ALLOCATION

The linear programming models described in this paper are designed to provide near-optimal allocations of weapons to maximize damage to an entire target system. They have recently been used by the Strategic Air Command in several force structure studies for the Joint Chiefs of Staff.

With the existing computer program, the models are capable of handling problems with as many as 30 different categories of targets and 50 different types of weapons. There is no limit on the number of targets in each category or the number of weapons of each type.

Although the model may be used independently, the Strategic Air Command is presently using it for the military target allocation in a two-sided, general war, total exchange model.

Strategic Retaliatory Scenario

In the scenario for this model, Red launches a first strike counter-force attack on Blue's offensive and defensive forces and command control elements. Red withholds sufficient forces to maintain a high proportion of Blue's urban-industrial resources in jeopardy. The goal of Red's first strike is to disarm Blue to such an extent that the outcome of any subsequent exchange will result in a decided balance of power favorable to the initiator.

The problem facing the Blue planner is to allocate his surviving forces to insure that he is still capable of destroying an unacceptable

fraction of the Red urban-industrial base. If possible he would also like to attack those forces, and their supporting facilities, that Red withheld with the aim of limiting possible damage to the Blue urban-industrial complex. The goal of the Blue response is to restore the balance of power with the hope of preventing the commitment of weapons on U/I targets.

Input Data

In the SAC model, a portion of the weapons which survive an enemy first strike are allocated to defense suppression, urban-industrial, and N^{th} power tasks first. Whatever is left over becomes the input to the military target problem. If the linear programming models are used independently, the available supply of weapons by number and type must be given.

When the defense suppression allocation has been made, the penetration rates (or, properly, the probabilities of successful penetration of enemy defenses) are determined. These figures in conjunction with the probabilities of launch survival, weapon system reliabilities, yield, accuracy, target hardness, etc., enable an expected probability of kill for each weapon/target combination to be computed.

While these figures represent much more than educated guesses, still they are uncertain. It has recently been suggested that instead of regarding these probabilities as single parameters, it would be better to think in terms of probability distributions of probabilities.[2] While this suggestion certainly has a considerable amount of intuitive appeal, the notion of expected value is probably at present the most manageable approach for aggregate analysis in support of future force structure determination.

To reduce the size of the military allocation problem to computable proportions, targets of similar worth and with approximately equal probabilities of survival when targeted with identical weapons, are aggregated into a single target category. For example, all missile silos

of a particular type in a particular geographic area might be placed in the same category.

Thus the input data consists of the number of weapons of each type, the number of targets in each category, and the matrix of probabilities of survival for each weapon/target combination.

Targeting Philosophy

The user of the model specifies three targeting controls.

First, the desired ratio of damage among the various categories is stated. Presumably, this would reflect the relative worths of the targets in each of the categories. This should not, however, be confused with assigning a point value to each category, as is done in most models. Using an allocation based on point value per target, the level of expected damage inflicted upon a particular target may have little relation to its worth. Frequently target hardness and weapon effectiveness become the driving factors. The user of such a model may find to his surprise that all his weapons went on soft targets of less worth simply because the model "scored more points" by such an allocation. The temptation at this point, of course, is to solve the problem again, but this time with an increased worth input for the harder targets. The sophisticated user realizes, however, that by so doing he has discarded the value system which he initially assumed was correct. He might just as well allocate his weapons subjectively without the benefit of a mathematical model.

On the other hand, in the method proposed here, there are no such surprises. If the user decides that he wants two categories of targets to survive an attack in the ratio of 2 to 3, the output reflects this. If a certain number of weapons are available, the actual allocation may result in the probabilities of survival (one minus the damage expectancy) being .2 and .3 respectively. If fewer weapons are available, the probabilities of survival might be .5 and .75 respectively, but in any case the probabilities of survival will be as small as the number of weapons committed allows and in the ratio specified.

Second, the maximum damage desired for each category is specified. This effectively stops allocation to a particular category at a point where the user considers further allocation either wasteful or for some other reason undesirable. The model may be used either with or without these restrictions.

Third, the user may declare certain target categories ineligible for certain weapon systems, because of range limitations, high expected attrition, or other reasons.

The Output

The output from the linear program gives the number of weapons of each type which are assigned to each category, and the average probability of survival of each category.

Sequenced Allocations

Rather than performing a single optimization in which all weapons available are used, it is sometimes desirable to make separate allocations for land launched missiles, sea launched missiles and bombers. By keeping the ratios the same from one allocation to another, it is possible to induce a considerable amount of cross-targeting. If the user wishes, however, he can vary the ratio in order to direct a certain class of weapons against various target categories. Furthermore, each allocation may be made completely in isolation or by considering the expected damage from prior attacks. This feature gives visibility to the overall degradation of effectiveness in case of a catastrophic failure of one or more classes of weapons.

Another interesting feature of sequenced allocations is the capability of assessing the value of reconnaissance followed by retargeting. In this user option, the number of targets within each category is reduced after each allocation by multiplying the original number of targets by the average probability of survival of the category.

Allocation Within Categories

After the initial master allocation has been made, it is known how many weapons of each type are to be assigned to each category. By using the integer programming model, described in the next chapter, it is then possible to determine the optimal assignment of weapons against each individual target within each category, should the user desire this information.

Variations

One of the virtues of linear (and integer) programming, is great flexibility. If additional operational restrictions on a problem are encountered, frequently it is possible to reflect such restrictions by minor modifications of the original program.

For example, if the user desires that every target be assigned at least one weapon, this constraint is easily added (assuming, of course, there are more weapons than targets). Also, if certain combinations of weapons are forbidden for operational reasons, a small change in the program eliminates such allocations from consideration.

CHAPTER II

THE LINEAR PROGRAMMING MODELS

MATHEMATICAL PRELIMINARIES

Allocations

Suppose that there are m distinct types of weapons and that there are b_i weapons of type i available $(i=1,\ldots,m)$. Also suppose that there are n categories of targets. Then by an allocation we mean an assignment of a certain number of weapons of each type to each category such that the available supply is not exceeded.

Let A_{ij} represent the number of weapons of type i assigned to category j. An allocation can be represented as the m by n matrix (A_{ij}) where the rows represent weapons types and the columns represent target categories.

For example, suppose that there are two weapon types, with fifteen weapons of the first type and ten weapons of the second type available, and three target categories. Then the following matrix represents one possible allocation:

$$\begin{bmatrix} 3 & 8 & 4 \\ 1 & 2 & 7 \end{bmatrix}$$

Notice that the rows add to 15 and 10 respectively, indicating that all weapons have been used. The matrix

$$\begin{bmatrix} 2 & 4 & 6 \\ 1 & 3 & 4 \end{bmatrix}$$

also represents an allocation, but in this case not all available weapons have been used. The matrix

$$\begin{bmatrix} 4 & 7 & 8 \\ 1 & 3 & 2 \end{bmatrix}$$

does not represent an allocation, since the sum of the entries in the first row exceeds 15.

The total number of possible allocations is extremely large even for rather small problems. For example, in a problem in which there are only two types of weapons, with three weapons of each type available, and two target categories, there are 400 possible distinct allocations. Thus, for realistic problems, it is computationally infeasible to measure the effectiveness of each possible allocation against the chosen criteria to determine which is optimal. More sophisticated procedures must be used to find optimal solutions.

Linear Programming

A linear programming problem is an optimization problem of the form:

$$\text{Maximize } C_1X_1 + C_2X_2 + \cdots + C_nX_n$$

Subject to

$$a_{11}X_1 + a_{12}X_2 + \cdots + a_{1n}X_n = b_1$$

$$a_{21}X_1 + a_{22}X_2 + \cdots + a_{2n}X_n = b_2$$

$$\cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot$$

$$a_{m1}X_1 + a_{m2}X_2 + \cdots + a_{mn}X_n = b_m$$

where the C_i, b_j, and a_{ij} are constants and the X_i are variables which may take on only non-negative values. (If, in addition, the X_i are constrained to be integers, such a problem is called an integer programming problem.)

The theory of linear programming is highly developed, and there are efficient algorithms for finding solutions to problems which either naturally occur in this form or which can be put in this form. The computational efficiency of linear programming algorithms results from the fact that every possible n-tuple $(X_1,\cdots,X_n)$ need not be checked in order to attain an optimal solution.

Computing Probability of Survival

If D_{ij} is the probability that a single target in the j^{th} target category will be destroyed by allocating a single weapon of type i to

it, then $S_{ij} = 1 - D_{ij}$ is the probability that it will survive.

If a single weapon of type k is also allocated to the target, then the probability that the target will be destroyed is $1 - (1 - D_{ij})(1 - D_{kj})$ and the probability that it will survive is $(1 - D_{ij})(1 - D_{kj}) = S_{ij} \cdot S_{kj}$, assuming that the arrival of one weapon is not dependent upon that of the other.

In general, then, if T_{ijk} weapons of type i are allocated to a particular target k in the $j^{\underline{th}}$ target category, the probability that it will survive is

$$S_{1j}^{T_{1jk}} S_{2j}^{T_{2jk}} \cdots S_{mj}^{T_{mjk}} = \prod_{i=1}^{m} S_{ij}^{T_{ijk}}$$

The average probability of survival of the $j^{\underline{th}}$ category is defined as the arithmetic mean of the probabilities of survival of each of the individual targets within the category, i.e.,

$$\frac{1}{a_j} \sum_{k=1}^{a_j} \prod_{i=1}^{m} (S_{ij})^{T_{ijk}}$$

where a_j is the number of individual targets in the $j^{\underline{th}}$ category.

For example, suppose two targets are aggregated into target category 1, and suppose one weapon of type 1 and two weapons of type 2 are allocated to the targets in this category. Suppose the actual allocation is one weapon of each type to the first target and one weapon of type 2 to the second target. If $S_{11} = .5$ and $S_{21} = .4$, then the probability of survival of the first target is $(.5)(.4) = .2$ and the probability of survival of the second target is .4. Thus the probability of survival of the category is $1/2(.2 + .4) = .3$.

An Approximation

The expression for the probability of survival of the $j^{\underline{th}}$ category,

$$\frac{1}{a_j}\sum_{k=1}^{a_j}\prod_{i=1}^{m}(S_{ij})^{T_{ijk}}$$

is difficult to deal with since it is a sum of products. Therefore we shall approximate it with

$$\prod_{i=1}^{m}(S_{ij})^{X_{ij}} \quad \text{where} \quad X_{ij} = \frac{1}{a_j}\sum_{k=1}^{a_j} T_{ijk}$$

In other words, X_{ij} is the average number of weapons of type i allocated per target to targets in category j. The rationale for this approximation follows.

$$\prod_{i=1}^{m}(S_{ij})^{X_{ij}} = \prod_{i=1}^{m}(S_{ij})^{\frac{1}{a_j}\sum_{k=1}^{a_j} T_{ijk}}$$

$$= \left[\prod_{i=1}^{m}(S_{ij})^{\sum_{k=1}^{a_j} T_{ijk}}\right]^{\frac{1}{a_j}}$$

$$= \left[\prod_{i=1}^{m}\left(\prod_{k=1}^{a_j}(S_{ij})^{T_{ijk}}\right)\right]^{\frac{1}{a_j}}$$

$$= \left[\prod_{k=1}^{a_j}\left(\prod_{i=1}^{m}(S_{ij})^{T_{ijk}}\right)\right]^{\frac{1}{a_j}}$$

At this point we notice that the terms $\prod_{i=1}^{m} (S_{ij})^{T_{ijk}}$ of the outside product are the probabilities of survival of each of the individual targets. Applying the well-known geometric-arithmetic inequality

$$\left[\prod_{i=1}^{n} P_i\right]^{\frac{1}{n}} \leq \frac{1}{n}\sum_{i=1}^{n} P_i$$

(with equality when all the P_i are equal) to our problem, we find that

$$\prod_{i=1}^{m} S_{ij}^{X_{ij}} \leq \frac{1}{a_j}\sum_{k=1}^{a_j} \prod_{i=1}^{m} (S_{ij})^{T_{ijk}}$$

In other words, $\prod_{i=1}^{m} (S_{ij})^{X_{ij}}$ is always less than or equal to the actual probability of survival of the $j^{\underline{th}}$ category, with equality occurring whenever the probability of survival of each individual target within the $j^{\underline{th}}$ category is the same.

The magnitude of the error introduced by making this approximation is difficult to assess for realistic problems, but it is believed to be small.

For example, suppose there are 25 targets in the $j^{\underline{th}}$ category, and suppose the probability of survival is 0.5 for the first 10, 0.4 for the second 10, and 1.0 for the remaining 5. Then the actual probability of survival of the category is 0.560 while the approximation yields 0.525.

THE LINEAR PROGRAMMING MODELS FOR ALLOCATION AMONG CATEGORIES

The underlying concept of the linear programming model is to force the probability of survival of each category to as small a value as possible consistent with: first, the targeting philosophy (which takes the form of a predetermined ratio of desired probabilities of survival

for each of the categories), and second, the available supply of weapons.

In the previous section a justification for the use of the expression

$$\prod_{i=1}^{m} (S_{ij})^{X_{ij}}$$

to approximate the probability of survival of the $j^{\underline{th}}$ category was given. The number of weapons of type i which are allocated is easily seen to be

$$\sum_{j=1}^{n} a_j X_{ij}$$

Suppose now that the targeting philosophy chosen indicates that the desired ratio of the probabilities of survival of the n categories are $C_1 : C_2 : \cdots : C_n$. Then if we introduce the auxiliary variable $\overline{Y}$, the mathematical model for the allocation problem is:

Minimize $\overline{Y}$

Subject to

$$1') \qquad \prod_{i=1}^{m} (S_{ij})^{X_{ij}} \leq C_j \overline{Y};\ j=1,\cdots,n$$

and

$$2') \qquad \sum_{j=1}^{n} a_j X_{ij} \leq b_i;\ i=1,\cdots,m$$

In rather imprecise language, the model uses $\overline{Y}$ with the coefficients C_j to "squeeze down" the probability of survival of each category in the stipulated ratio consistent with the available supply of weapons.

This mathematical model is, however, a <u>non-linear</u> programming problem since conditions 1' are non-linear in the variables X_{ij}. This difficulty is easily resolved by taking the logarithm of both sides of each inequality, yielding the equivalent set of inequalities

$$1)\ \sum_{i=1}^{m}(\log S_{ij})X_{ij} \leq \log C_j + \log \bar{Y};\ j=1,\cdots,n$$

Now let $Y = \log \bar{Y}$. By the monotonicity of the logarithm function, minimizing Y will minimize $\bar{Y}$. This change yields an equivalent linear programming problem for which solution methods are readily available.

The Basic LP Model

The basic linear programming model is

Minimize Y

Subject to

$$1.)\sum_{i=1}^{m}(\text{Log } S_{ij})X_{ij} - Y \leq \log C_j;\ j=1,\cdots,n$$

and

$$2.)\sum_{j=1}^{n}a_jX_{ij} \leq b_i;\ i=1,\cdots,m$$

Example:

Consider an allocation problem with three weapon types and three target categories.

Let

$a_1 = 10$ $\quad$ $b_1 = 20$

$a_2 = 16$ $\quad$ $b_2 = 25$

$a_3 = 21$ $\quad$ $b_3 = 28$

$S_{11} = .20$ $\quad$ $S_{12} = .17$ $\quad$ $S_{13} = .25$

$S_{21} = .30$ $\quad$ $S_{22} = .29$ $\quad$ $S_{23} = .34$

$S_{31} = .45$ $\quad$ $S_{32} = .46$ $\quad$ $S_{33} = .49$

when a_j is the number of targets in category j, and S_{ij} is the probability of survival of a single target in category j when targeted with a single weapon of type i, and let the desired ratio of target category survivability be 100:94:87. Then the Basic LP model is

Minimize Y

Subject to

$(\log .20)X_{11}+(\log .30)X_{21}+(\log .45)X_{31}-Y\leq \log 100$

$(\log .17)X_{12}+(\log .29)X_{22}+(\log .46)X_{32}-Y\leq \log 94$

$(\log .25)X_{13}+(\log .34)X_{23}+(\log .49)X_{33}-Y\leq \log 87$

$10X_{11}+16X_{12}+21X_{13}\leq 20$

$10X_{21}+16X_{22}+21X_{23}\leq 25$

$10X_{31}+16X_{32}+21X_{33}\leq 28$

Interpretation of Results

The output of the Basic LP model is the set of values for the variables X_{ij} which minimize Y and satisfy the constraint inequalities. For the previous example, these values were found to be:

$X_{11} = .429$	$X_{12} = .981$	$X_{13} = 0$
$X_{21} = 0$	$X_{22} = 0$	$X_{23} = 1.190$
$X_{31} = 1.234$	$X_{32} = 0$	$X_{33} = .745$

Since X_{ij} is defined to be the number of weapons of type i allocated per target to targets in category j, the actual number of weapons allocated is a_jX_{ij}. For example, the number of weapons of type 3 allocated to category 3 is $(21)(.745) = 15.6$. Since this number is not an integer, it should be rounded to the next lower integer. This rounding error is quite small, percentagewise, if the value for a_jX_{ij} is large.

Using our approximation for the probability of survival of each category, the probabilities in the example are .187, .176, and .163 for categories one, two, and three respectively. These are, of course, precisely in the ratio 100:94:87. The actual probability of survival of each category can be computed only after the allocation within each category is made. An optimal method for doing this will be discussed later in this chapter.

The Advanced LP Model

In realistic targeting problems, it is frequently desirable to limit the level of destruction of a target system to a certain predetermined figure rather than simply to use all available weapons. This can be achieved through an extension of the Basic LP model by adding a constraint for each "cutoff" desired and by including additional auxiliary variables.

To insure that the probability of survival of the j^{th} target category is at least a certain predetermined value P_j, we add the constraint

$$\sum_{i=1}^{m}(\log S_{ij})X_{ij} \geq \log P_j$$

for each target category j where such a constraint is desired.

Unfortunately, the addition of these constraints is not all that is necessary. Since from the Basic LP model we already have the constraint

$$\sum_{i=1}^{m}(\log S_{ij})X_{ij} - Y \leq \log C_j$$

the two together imply the constraint

$$\log P_j - \log C_j \leq Y$$

In other words, when cutoff is achieved on one category, Y could be minimized no further and the allocation would stop at that point.

To eliminate this problem, more than one auxiliary variable must be introduced. Instead of Y, the auxiliary variables, $Y_1, Y_2, \cdots, Y_n$ are used.

The Advanced LP Model is

Minimize $\quad m_1Y_1+m_2Y_2+\cdots+m_nY_n$

Subject to

1) $\quad \sum_{i=1}^{m}(\log S_{i1})X_{i1} - Y_1 \leq \log C_1$

$$\sum_{i=1}^{m}(\log S_{i2})X_{i2} - Y_1 - Y_2 \leq \log C_2$$

- - - - - -

$$\sum_{i=1}^{m}(\log S_{in})X_{in}-Y_1-Y_2-\cdots-Y_n\leq C_n$$

2) $$\sum_{j=1}^{n}a_jX_{ij}\leq b_i, i=1,\cdots,m$$

3) $$\sum_{i=1}^{m}(\log S_{ij})X_{ij}\geq \log P_j;\ j=1,\cdots,n$$

where the inequalities are ordered so that $\frac{C_1}{P_1}\leq\frac{C_2}{P_2}\leq\cdots\leq\frac{C_n}{P_n}$ and $m_1,m_2,\cdots,m_n$ are suitably chosen positive numbers such that $m_1>m_2>\cdots>m_n$.

If m_1 is chosen sufficiently large relative to the other m_i, then until the first cutoff is reached, Y_1 will be the only Y_i which is reduced in value. Up to this point the model is for all practical purposes identical with the Basic LP Model.

However, when the first cutoff is achieved, Y_1 can be reduced no further. At that point if m_2 is sufficiently large relative to the remaining m_i, then until the next cutoff is achieved, Y_2 will be the only Y_i which is reduced in value, and so on.

The choice of a proper set of m_i parameters is a difficult one, and a satisfactory method for choosing it remains an unsolved problem. In realistic problems it appears that if m_i is greater than m_{i+1} by a factor of ten, this is sufficient. However for an allocation problem with a large number of cutoffs, and hence a large number of m_i, scaling problems within the computer occur. Research in determining the minimum factor by which each m_i must differ from its successor is continuing.

Example (Continued):

If, in the previous example, it is desired to limit destruction of each category so that its probability of survival is at least .168, then the Advanced LP Model is:

Minimize $9Y_1+3Y_2+Y_3$

Subject to

$(\log .25)X_{13}+(\log .34)X_{23}+(\log .49)X_{33}-Y_1 \leq \log 87$

$(\log .17)X_{12}+(\log .29)X_{22}+(\log .46)X_{32}-Y_1-Y_2 \leq \log 94$

$(\log .20)X_{11}+(\log .30)X_{21}+(\log .45)X_{31}-Y_1-Y_2-Y_3 \leq \log 100$

$10X_{11}+16X_{12}+21X_{13} \leq 20$

$10X_{21}+16X_{22}+21X_{23} \leq 25$

$10X_{31}+16X_{32}+21X_{33} \leq 28$

$(\log .25)X_{13}+(\log .34)X_{23}+(\log .49)X_{33} \geq \log .168$

$(\log .17)X_{12}+(\log .29)X_{22}+(\log .46)X_{32} \geq \log .168$

$(\log .20)X_{11}+(\log .30)X_{21}+(\log .45)X_{31} \geq \log .168$

Here a factor of 3 for the m_i turned out to be sufficient. The values for the X_{ij} are

$X_{11} = .401$	$X_{12} = .999$	$X_{13} = 0$
$X_{21} = 0$	$X_{22} = 0$	$X_{23} = 1.190$
$X_{31} = 1.331$	$X_{32} = 0$	$X_{33} = .699$

Using the approximation for the probability of survival of each category, these values are .181, .171, and .168 for categories one, two, and three respectively. In comparing these results with the results of the example in which no cutoffs were included, it can be seen that weapons were taken off category three to bring its probability of survival up to .168 and were applied to categories one and two to reduce their probabilities of survival in the specified ratio.

The Alternate Advanced LP Model

If suitable parameters $m_1,\cdots,m_n$ cannot be found, then a succession of Basic LP problems can be solved to find the solution in the following way:

Write the problem as a Basic LP problem with the one additional constraint

$$\sum_{i=1}^{m}(\log S_{i1})X_{i1} \geq \log P_1$$

If the allocation results in this constraint being satisfied as a strict inequality, then no cutoffs were achieved and the solution to the original problem has been attained.

If, on the other hand, an equation results, the same Basic LP problem must be solved with the two constraints

$$\sum_{i=1}^{m}(\log S_{i1})X_{i1} = \log P_1$$

$$\sum_{i=1}^{m}(\log S_{i1})X_{i1} \geq \log P_2$$

added and the constraint

$$\sum_{i=1}^{m}(\log S_{i1})X_{i1} - Y \leq \log C_1$$

removed.

The succession of modified Basic LP problem is continued until the newly added inequality is satisfied strictly. If this never happens, then the "sufficient weapons condition", to be described next, has occurred.

The "Sufficient Weapons Condition" LP Model

If in a particular allocation problem in which cutoffs are included for each category there are a sufficient number of weapons available to achieve each cutoff, then it seems appropriate to use a different criterion for making the allocation. This may take the form of minimizing the total number of weapons used to do the job, using some weapons in preference to others, or some similar criterion. The LP Model in this case is

Minimize

$$\sum_{j=1}^{n}\sum_{i=1}^{m} d_i a_j X_{ij}$$

Subject to

1) $$\sum_{j=1}^{n} a_j X_{ij} \leq b_i;\ i=1,\cdots,m$$

2) $$\sum_{i=1}^{m} (\log S_{ij}) X_{ij} \leq \log P_j;\ j=1,\cdots,n$$

where the d_i are suitable chosen positive constants.

If it is desired to minimize the total number of weapons used, all the d_i are set equal to one. On the other hand, if it is desired to use weapon i in preference to weapon j, then d_j is chosen greater than d_i. This has the effect in the LP Model of making it more "expensive" to allocate weapons of type j. Clearly many other modifications are possible depending on the desires of the user.

Example (Continued):

If in the previous example, the cutoffs had been set at .25 for each category, then the sufficient weapons condition would exist. Thus for our example, if it is desired to minimize the number of weapons, we must

Minimize $10X_{11}+16X_{12}+21X_{13}+10X_{21}+16X_{22}+21X_{23}+10X_{31}+16X_{32}+21X_{33}$

Subject to

$10X_{11}+16X_{12}+21X_{13} \leq 20$

$10X_{21}+16X_{22}+21X_{23} \leq 25$

$10X_{31}+16X_{32}+21X_{33} \leq 28$

$(\log .20)X_{11}+(\log .30)X_{21}+(\log .45)X_{31} \leq \log (.25)$

$(\log .17)X_{12}+(\log .29)X_{22}+(\log .46)X_{32} \leq \log (.25)$

$(\log .25)X_{13}+(\log .34)X_{23}+(\log .49)X_{33} \leq \log (.25)$

The values for the X_{ij} are

$X_{11} = .748$	$X_{12} = .782$	$X_{13} = 0$
$X_{21} = 0$	$X_{22} = 0$	$X_{23} = 1.190$
$X_{31} = .228$	$X_{32} = 0$	$X_{33} = .142$

Each of the categories has a probability of survival of 0.25. Since weapon type three is the least effective, weapon types one and two were used until their supply was exhausted and then five weapons of type three were used to bring the probability of survival down to the specified figure.

If, for some reason, it is desired to use weapon type three with highest priority, and then two and one in that order, this can be accomplished by setting $d_1=100$, $d_2=10$, $d_3=1$ in the objective function and leaving the constraints unchanged. In this new problem, the X_{ij} are

$$X_{11} = 0 \qquad X_{12} = .563 \qquad X_{13} = 0$$
$$X_{21} = 1.15 \qquad X_{22} = .313 \qquad X_{23} = .403$$
$$X_{31} = 0 \qquad X_{32} = 0 \qquad X_{33} = 1.333$$

In this case, all weapons of types two and three are used and eleven weapons of type one are spared.

Other Constraints

One of the strengths of linear programming is the capability of adding additional constraints to reflect different operational considerations without abandoning the entire model.

For example, if it is desired that every target in category j be targeted with at least one weapon, then the addition of the following constraint will assure that the final allocation reflects this requirement:

$$\sum_{i=1}^{m} X_{ij} \geq 1.0$$

Also, if a weapon of type i is restricted from hitting targets in category j, then letting $S_{ij}=1$, effectively prohibits such an allocation.

Clearly, many other variations are possible depending upon the requirements of the particular allocation problem.

THE INTEGER PROGRAMMING MODEL FOR ALLOCATION WITHIN A CATEGORY

After it has been determined how many weapons of each type should be allocated to each category, it is necessary to determine how these weapons should be distributed among the individual targets in that category so as to minimize the average probability of survival of the targets in the category.

In the case where the total number of weapons assigned to a category does not exceed the number of targets in the category, the minimum value for the probability of survival of the category is attained when only one weapon is assigned to each target to the extent that they are available.

However, when the number of weapons allocated to a category exceeds the number of targets in that category, the problem of allocating them optimally becomes more complex. The integer programming model which follows is designed to insure an optimal allocation in this case.

Notation

N = number of targets in the category

L = maximum permissible number of weapons assigned to any individual target in the category

K = number of different weapon types assigned to the category

b_j = number of weapons of type j assigned to the category; $j=1,\cdots,K$

S_j = probability of survival of a single target in the category when targeted with a single weapon of type j.

A = number of different allocations

a_i = $(a_{i1}, a_{i2}, \cdots, a_{iK})$ = allocation of type i, where a_{ij} is the number of weapons of type j used in allocation i; $i=1,2 \cdots ,A$

P_i = probability of survival of an individual target with an allocation of type i.

T_i = number of individual targets which are targeted with an allocation of type i.

The Allocations

As mentioned above, an allocation against an individual target is defined to be a k-vector in which the j^{th} component is the number of

weapons of type j used in the allocation. Since the maximum number of weapons assigned to any individual target is L, the allocation is further restricted so that the sum of the components is L or less.

Given a value for L and K, the number of different allocations is

$$A=\sum_{i=o}^{L}\binom{k+i-1}{i}$$

Associated with each allocation a_i is a number P_i, representing the probability of survival of an individual target, when targeted with allocation a_i.

Example

Suppose $K=L=2, S_1=.3, S_2=.5$

$$\text{Then } A=\sum_{i=o}^{2}\binom{2+i-1}{i}=\binom{1}{0}+\binom{2}{1}+\binom{3}{2}=6$$

The six allocations are:

$a_1 = (0,0)$	$a_3 = (0,1)$	$a_5 = (0,2)$
$a_2 = (1,0)$	$a_4 = (2,0)$	$a_6 = (1,1)$

The six associated probabilities of survival are

$P_1=(.3)^0(.5)^0=1$	$P_3=(.3)^0(.5)^1= .5$	$P_5=(.3)^0(.5)^2= .25$
$P_2=(.3)^1(.5)^0= .3$	$P_4=(.3)^2(.5)^0= .09$	$P_6=(.3)^1(.5)^1= .15$

<u>The Integer Programming Problem</u>

The integer programming problem which minimizes the average probability of survival of targets in the category subject to weapon availability is:

$$\text{Minimize} \quad \sum_{i=1}^{A} P_i T_i$$

Subject to

$$1.) \qquad \sum_{i=1}^{A} T_i = N$$

2.) $\sum_{i=1}^{A} a_{ij}T_i \leq b_j;\ j=1,\cdots,K$

If N=400, b_1=350, and b_2=220, then the integer programming problem is:

Minimize $1T_1+.3T_2+.5T_3+.09T_4+.25T_5+.15T_6$

Subject to

1.) $T_1+T_2+T_3+T_4+T_5+T_6=400$

2.) $0T_1+1T_2+0T_3+2T_4+0T_5+1T_6 \leq 350$

$0T_1+0T_2+1T_3+0T_4+2T_5+1T_6 \leq 220$

Further Refinements

In linear and integer programming problems, the most common way of adding restrictions to a problem is to add appropriately chosen constraints. In this problem, however, some interesting restrictions may be obtained by eliminating certain variables.

1. Cross-Targeting If it is desired that each target be hit by more than one different weapon type, eliminate all variables which are associated with allocations which do not meet this requirement. Another way to do this is to retain the variable, but set the coefficient of this variable equal to one in the objective function.

2. In General All allocations may be examined to see that they meet operational requirements. Those which do not are eliminated, and the answer obtained to the resulting problem reflects these restrictions.

CHAPTER III

THE COMPUTER MODEL

GENERAL DESCRIPTION

A weapon allocation program (WALLOP) using linear programming to allocate weapons to the objective military targets is presently interfaced at Hq SAC with other war gaming computer models. These programs are merely bookkeeping models that build the weapon, target, and damage expectancy arrays to be used as inputs for the LP models.

Problem Size

The program in its current configuration has the capability of handling 30 target categories and 50 weapon types. In this program all weapons of identical characteristics are grouped into types: e.g., all MINUTEMAN II would be one type, and targets of identical, or very similar, characteristics are grouped into categories. This grouping causes very little loss in accuracy and considerably shortens computational time. There is no restriction on the number of targets in each category or number of weapons in each weapon slot.

Running Time

Typical cases using 29 target categories and 13 weapon types have been solved in approximately 40 seconds of CPU time on the IBM dual core 7090 computer.

Attack Sequence

Instead of allocating all weapons at one time, it is of interest to the user/analysts at Hq SAC to allocate the weapons in a sequence of attacks. A sequence of four different attacks are presently structured in the program for each input case.

1. An ICBM Attack
2. An SLBM Attack

3. A Bomber Attack

4. A Bomber Attack when bomber weapons are restricted from certain target categories.

The weapon laydown for each attack is optimized using one of the LP models. It takes only a very minor coding modification to reconfigure the forces in each attack or, if desired, to allocate all the weapons at one time.

Program Input

The input to WALLOP consists of:

1. The target, weapon, and damage expectancy matrices. They may be generated by other models and then interfaced with the program by the use of magnetic tape as done at Hq SAC, or may be read in from input cards.

2. The relative importance of the target categories for each attack. One may specify either ratios for aggregate entries (grouping the target categories in hard and soft targets) or ratios for individual categories. Both modes may be freely intermixed.

3. The philosophy of weapon allocation. The philosophy of weapon allocation is related to the utilization of the achieved damage expectancy (DE) between successive attacks.

a. Isolation. When targeting in isolation, each weapon system attacks the entire target complex taking no credit for any previous damage achieved, hence a very conservative targeting philosophy. Also, this option may be used to ensure cross targeting to reduce the risks of unexpected high failure rates of any specific weapon type, or group of weapon types, thus avoiding a catastrophic failure of the war plan.

b. Time Sequence. In this option the attacks are structured into the program as indicated previously. The missile attack was broken into an ICBM and an SLBM portion so that the study director or user/ analyst may have better visibility of the contribution by each weapon system to the total damage achieved. The ICBM and SLBM attacks precede the bomber attack as would normally be the case due to their short reaction time. The ratios of the desired damage to the individual

categories are adjusted by the damage achieved in the individual target categories during the previous attacks. It is an easy modification to the present program to adjust the desired ratios of damage by a specified percentage of the damage achieved in all the categories. The mathematical development of this option is presented in Appendix.

c. Reconnaissance Mode. This option is designed so that the study director or user/analyst can see the effect of having reconnaissance. The attacks are structured in the time sequence mode. After each attack, the number of targets in each category is adjusted by the ones destroyed. Subsequent attacks utilize the remaining targets in each category. This mode could be used to simulate having empty silo information, for example. The forces would have to be sufficiently flexible and responsive to permit target changes indicated by the reconnaissance.

Computer System

The program WALLOP is currently implemented on the dualcore SAC IBM 7090 computer. Core Storage consists of two 32,768 word modules designated A and B. Core B is a 32K word storage bank identical to Core A. The additional core storage feature gives the system a 65K capability. Unfortunately, all FORTRAN users are restricted from using B core because the IBSYS operating system is unable to address this core bank. The maximum address that can be contained in the 7090 instruction is 77777_8, which covers the entire address range of one core bank. Several manual and internal switches may be checked by the control unit to determine which core unit is actually being referenced. Subroutines were written in MAP to be called from the FORTRAN main program which facilitates the exchange of data between storage modules Core A and Core B. The subroutines GET and PUT were designed to give the FORTRAN programmer full use of "B" core as a supplementary storage area. Only one or more consecutive core locations may be moved through each call of these subroutines. Due to the size of each LP problem and the number of cases investigated for each study, these subroutines made the entire linear

programming allocation scheme possible in an acceptable amount of computer running time. The only restriction remaining is that all data processing must be performed in "A" core.

Program Restrictions

Just two major constraints have been programmed into the program WALLOP. First, WALLOP can only handle a maximum of 100 constraint equations, which could be increased by enlarging internal array storage. Second, the number of constraint equations times the number of variables may not exceed 32760.

DETAILED DESCRIPTION

Input Processing

The program WALLOP will build the entire input necessary for the LP models from the four arrays: targets, weapons, damage expectancy, and desired ratios. Figure 1 depicts the general interrelationship between the input to the program and input necessary for the LP weapon allocation models.

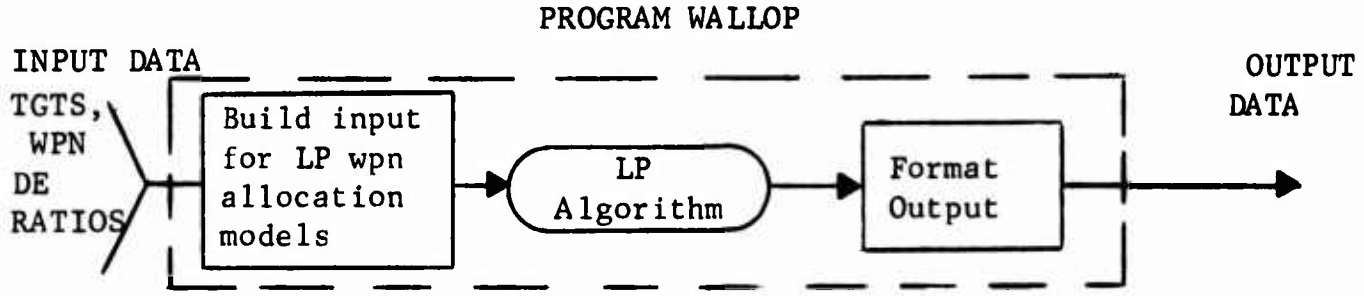


Figure 1

Maximum usage of default options have been utilized for user convenience and in order to keep input cards at minimum.

Parameter cards may be inserted into the input stream to override any default characteristic. For example, at SAC, the target, weapon, and DE matrix are created by other computer programs and are written on magnetic tape. The default option is to process all the cases on this tape unless a "case" card is detected in the input stream of control

cards. Only the cases indicated on this "case" card will be processed. The parameter cards may appear in ANY order and are processed by the subprogram INPUT. Input cards may be used to alter all arrays of the original input data when interfacing with magnetic tape.

The major input arrays, size and contents are:

TGT (30)	The number of targets in each category; each entry is considered as one category.
WEAP (50)	Fifty different weapon types may be used.
SIJ (50,30)	The survivability of each weapon/target category combination $(1 - D_{ij})$
VAL (30)	The ratios of damage by category for the current attack.
MINS (30)	The minimum probability of survivability for each category--cut off.
TMAP (30)	The target category mapping vector which contains the index number of the categories containing targets for this attack.
WMAP (50)	The weapon type mapping vector contains the index number of the weapons for the current attack. (Explained below.)
NCAT	The number of target categories for this attack.
NWEAP	The number of different weapon types for this attack.

The interrelation of the arrays are presented pictorially in Figure 2.

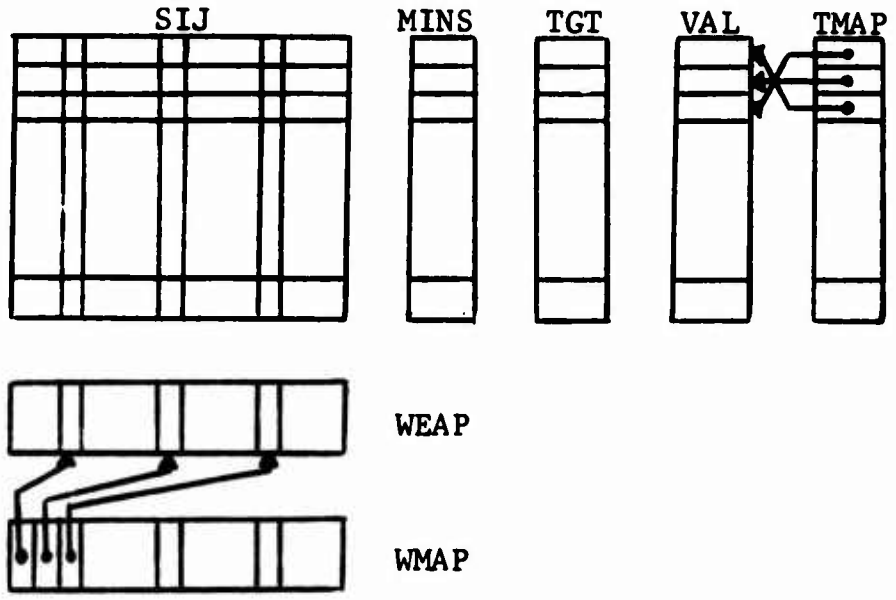


Figure 2

To allow complete freedom in the choice of target categories and weapon types, all access to the data arrays is made through their respective mapping vectors. Hence, for example, if only five weapon types are to be used, ANY five slots may be used in the WEAP array. The mapping arrays for weapons, (WMAP) will be built with the first five entries containing the index of the weapon slots used.

The advanced LP algorithm specifies the order of equation so that

$$\frac{C_1}{P_1} \leq \frac{C_2}{P_2} \leq \cdots \leq \frac{C_n}{P_n}$$

holds. These C_i may be different in each attack, hence the order of equations may also be different.

All equations will be generated by using the entries in TMAP. TMAP contains the index of the valid target entries of the 30 possible categories which is also the index of the corresponding values for the desired ratios, and when applicable, the minimum level for the probability of survival (cut-off). By forcing all entries to any of the arrays through the appropriate mapping vector, the number of words to be sorted is reduced from 159 to 3 words for each exchange. Hence, the time required for the sort for each attack is reduced.

Example:

The contents of TGT (7) is to be interchanged with TGT (25). Not only must the entries in TGT, VAL, and the MINS array be interchanged but also the 50 words for each entry in the SIJ array must be interchanged accordingly.

Sort Procedure

A modified version of the bubble sort is utilized because it is very efficient in the use of core storage required. One drawback to it is that it requires a rather large number of operations. It seemed to be unnecessary to go to a higher level of sophistication in a sort technique for a maximum of 30 items. The principle of the sort is to

start at the bottom of the argument file and compare pairs of arguments in turn. The larger argument of a pair is placed in the lower index position of the pair; the smaller argument value is placed in the higher index position.

The comparison continues until element X_0 has been processed. The result is that the largest argument in the file has "bubbled" to the top position. The bubbling procedure is then repeated on the resulting file reduced by the top member which need not be processed any longer. The second pass bubbles the second highest value to its proper position on the file. After n-1 passes, a file of n items will be sorted.

It was recognized that for certain files of data, the file may be sorted after fewer than the worst case of n passes. An indicator variable was introduced and set to 0 at the start of each pass and set to 1 after each bubble interchange. After each pass is complete, a test of the value of this indicator will determine if a subsequent pass is necessary.

Generation of Arrays A and B

Since "B-core" cannot be used for processing under the IBSYS operating system, it became the logical place to store Array A, the coefficients of the constraint equations. All 32K is reserved for Array A. All data must be in consecutive locations to be transferred to or from "B-core" through one call of the subroutines GET and PUT which do the actual data transfer. Since the LP algorithm utilizes Array A column by column, it is very desirable to build up Array A column by column for most efficient data transfer between core banks. To allow rapid identification of the beginning core location of each column in B-core, each column of the Array A has a specific location in B-core dependent upon the column number and the number of constraint equations in the problem. This in effect packs all the data in B-core and increases the problem size that can be handled.

As a specific column is requested by the LP algorithm by column number, it is retrieved from B-core using $1 + M^*(J-1)$, to locate its

starting core location in B-core, where M is the number of constraint equations and J is the desired column number.

Figure 3 portrays how Arrays A and B appear for a case with 2 weapon types and 3 targets for the basic and advanced LP model.

BASIC LP MODEL

	ARRAY A VARIABLES							ARRAY B
	X_{11}	X_{12}	X_{13}	X_{21}	X_{22}	X_{23}	Y_1	B
RATIO EQUATIONS	$\ln S_{11}$			$\ln S_{21}$			1	C_1
		$\ln S_{12}$			$\ln S_{22}$		1	C_2
			$\ln S_{13}$			$\ln S_{23}$	1	C_3
STOCKPILE EQUATIONS	A_1	A_2	A_3					b_1
				A_1	A_2	A_3		b_2

ADVANCED LP MODEL

	X_{11}	X_{12}	X_{13}	X_{21}	X_{22}	X_{23}	Y_1	Y_2	Y_3	B
RATIO EQUATION	$\ln S_{11}$			$\ln S_{21}$			1	1	1	C_1
		$\ln S_{12}$			$\ln S_{22}$		1	1		C_2
			$\ln S_{13}$			$\ln S_{23}$	1			C_3
STOCKPILE EQUATIONS	A_1	A_2	A_3							b_1
				A_1	A_2	A_3				b_2
CUT-OFF EQUATION	$-\ln S_{11}$			$-\ln S_{21}$						M_1
		$-\ln S_{12}$			$-\ln S_{22}$					M_2
			$-\ln S_{13}$			$-\ln S_{23}$				M_3

FIGURE 3

where c_1, c_2, c_3 are the desired ratios.

b_1, b_2, b_3 are the total number of weapons.

M_1, M_2, M_3 are the minimum levels of survival.

A_1, A_2, A_3 are the number of targets in each category.

$S_{11}, \cdots, S_{23}$ are the probabilities of survival.

All terms in Arrays A and B are generated by addressing through TMAP, the target mapping vector which has been sorted so that the order of equations for the advanced LP model will hold. Looking at Array A in Figure 3, one sees that the terms for the cut-off equations are just the negatives of the terms in the ratio equations. The program will build one entire column including the cut-off term but will only transfer to B-core the appropriate number of words that correspond to the model selected by the user.

Objective Function

A major difficulty was encountered in determing the numeric values for the co-efficients of the Y_i in the objective function for the advanced LP model. The LP algorithm would loop, give non-optimum answers, or the ratios were incorrect for the probability of survival. This was attributed to errors due to round off, truncation, and scaling which would cause weapons to be misallocated.

Different schemes were devised to determine the value of these co-efficients to alleviate this problem. The program now builds these co-efficients iteratively.

$c_1 = 1$

$c_{i+1} = E * c_i$

Where $E = 10$

While the value of E is an ad hoc value, it has worked reasonably well for most problems. The user/analyst may override this value of E by a parameter card if desired.

Linear Programming Algorithm

The program WALLOP will call an LP algorithm in the form of a subroutine to solve the weapon allocation problem after all arrays have been built. The desirable features of any LP algorithm are:

1. Accuracy of the solution, including its independence of row and column scale factors
2. Storage needed for data
3. Core storage necessary for the subroutine
4. Speed of the solution

An LP algorithm originally written by RAND was utilized after extensive modification to run on the 7090 and to utilize B-core efficiently.[1] The method of inverting used in this routine is of interest since it differs from the usual procedures and is presented here.

Inversion Method

A typical method of inverting is to pivot in each column that is part of the basis, pivoting in the row in which the column entry has the largest absolute value. For example, suppose

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 1 \\ \epsilon \end{pmatrix}$$

where ϵ is a small number whose absolute value is less than half the accuracy to which unity can be represented. Thus, on an IBM 7090, ϵ might be 10^{-9}. Now pivoting in the first column, we would choose the first row and obtain:

$$\overline{A} = \begin{pmatrix} 1 & \frac{1}{2} \\ 0 & -\frac{1}{2} \end{pmatrix} \qquad \overline{B} = \begin{pmatrix} \frac{1}{2} \\ \epsilon - \frac{1}{2} \end{pmatrix}$$

But we assume that ϵ is small enough so that $\epsilon - \frac{1}{2}$ numerically becomes $-\frac{1}{2}$. Then we pivot in the second row of the second column and obtain:

$$\bar{A} = I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \bar{B} = W = \begin{pmatrix} 0 \\ 1 \end{pmatrix} ;$$

hence, $w_1 = 0$, $w_2 = 1$. We note the correct answer is $w_1 = \epsilon$, $w_2 = 1-2\epsilon$. The ϵ has been absorbed into the round-off error.

The following pivot scheme avoids this problem:

1) Let $y_1, y_2, \cdots, y_m$ be the transformed column in which we wish to pivot. Let $\bar{b}_1, \bar{b}_2, \cdots, \bar{b}_m$ be the current values of the transformed constant vector (i.e., what was previously called w_i).
2) Let S be the set of integers such that $i \epsilon S$ if $|y_i| > TP$ where TP is the pivot tolerance.
3) Let T be the set of integers such that $i \epsilon T$ if $\bar{b}_i = 0$.
4) If $S \cap T$ is not vacuous, do Step A; if $S \cap T$ is vacuous, do Step B.
   A) Let IR, $IR \epsilon S \cap T$, be an integer such that

      $$|y_{IR}| \geq |y_i| \text{ for all } i \epsilon S \cap T.$$

   B) Let IR, $IR \epsilon S$, be an integer such that

      $$\left|\frac{y_{IR}}{\bar{b}_{IR}}\right| \geq \frac{y_i}{\bar{b}_i} \text{ for all } i \epsilon S.$$

Thus we are effectively pivoting in the row that has the largest ratio $|y_i/\bar{b}_i|$. In the above example, the largest ratio in the first column is the second row; hence the result of the first pivot is

$$\bar{A} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \bar{B} = \begin{pmatrix} 1-2\epsilon \\ \epsilon \end{pmatrix}$$

Here the $1-2\epsilon$ will be rounded, presumably, to unity. Then we pivot in the first row of the second column to obtain:

$$\bar{A} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \bar{B} = \begin{pmatrix} 1 \\ \epsilon \end{pmatrix}$$

Hence we get $w_2 = 1$, $w_1 = \epsilon$, which is actually the correct answer within round-off error. This method has the disadvantage that, in not choosing the largest element in a column to determine the pivot row, we may get an element that barely exceeds the pivot tolerance. But we nonetheless believe that the increased accuracy justifies this approach.

Other choices could have been made for the pivot element. For example, instead of pivoting on columns in column order, one could first choose the eligible row with the smallest w_j, then choose from that row of the matrix the eligible element with the largest absolute value. This was not done because of timing considerations, and because it would make the selections scale-dependent.

The pivot tolerance TP is used in Step 2 to determine whether or not a y_i is zero. A satisfactory method of computing this tolerance is first to compute, on every iteration,

$$\text{YMAX} = \max_{i=1}^{m} |y_i|$$

Then the pivot tolerance for <u>that iteration</u> is taken to be TP=YMAX * 2^{-16}. Then, on a machine that carries numbers in floating point notation to a relative accuracy of 2^{-27}, we assume $y_i = 0$ if $|y_i| \leq$ TP.

<u>Output</u>

Two total damage figures are computed. One includes bombers unrestricted, and the other includes a bomber attack in which the bombers are restricted from specific target categories. The damage achieved by each system in isolation is also shown so that the study director or user/analyst can weigh the contribution for each weapon system in relation to the total.

When using the advanced LP model, the program will check the probability of survival to see if the desired ratios have been achieved. If they do not conform to the desired ratios, a scaling problem is present. An error message is printed and the attack is rerun without cut-off.

Since the probability of survival is an approximation, an integer programming routine must be used to find the actual P_s. For the insufficient weapon case where there are more targets than weapons allocated, the actual P_s can be calculated. The approximate P_s will be used in place of the actual P_s for the other case until an integer programming routine can be implemented into the program. The program will use an iterative procedure to adjust the desired ratios in order that the actual P_s corresponds to the desired ratio of P_s. However, convergence may require too many iterations to be practical and may not be possible at all. For all insufficient weapon cases, experience has shown the resultant allocation to be very good (if not precisely optimal) often after only a few iterations. During the iterative process, the program will take the solution for one cycle as the initial basis for the next cycle, hence, computer time is kept at an absolute minimum.

General Overview

This completes one cycle through the program for one attack. WALLOP will then process the next attack by setting the desired weapons in the weapon mapping array (WMAP) and the desired targets in the target mapping array (TMAP). The entire cycle is then repeated.

A general flow diagram is depicted in Figure 4.

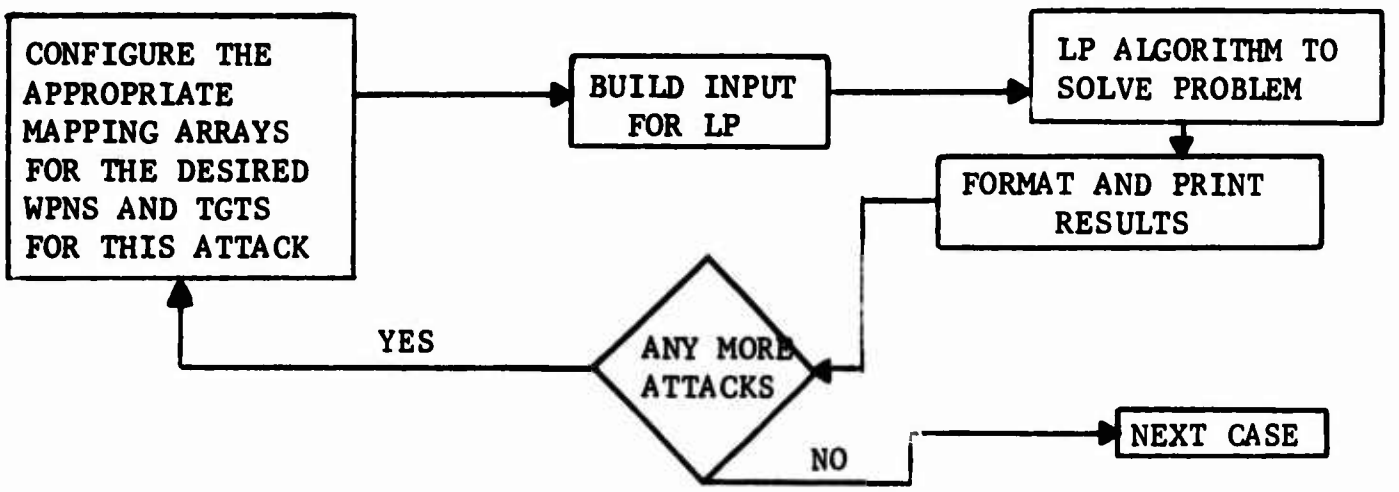


FIGURE 4

A more detailed flow diagram is depicted in the Appendix.

FUTURE CONFIGURATIONS

The following functional flow chart is presented to illustrate the future characteristic of the program WALLOP. The weapon minimization models are currently undergoing testing, hence are not included in this report. The only major building block still not implemented is the integer programming models for allocating within a category. An integer programming routine must still be implemented on the dual core IBM 7090 at SAC.

Since the initial implementation of the program, routines have been designed, coded and implemented to enable FORTRAN users to use disk storage. A version of the program, which builds the constraint equations row by row on the disk in contrast to the present column by column approach, is currently under test. Array A is then recalled into A core in blocks of 100 x 100 words. The matrix is transformed ($A \rightarrow A'$) and written into B-core for use of the LP algorithm. By building the constraint equations as defined, future constraints can be added rather easily. It is currently planned for the model to be modified so that parameter cards will control all constraints. For example, if coverage is wanted on a particular category, a parameter card will specify this requirement and the constraint equation will be included in the LP model.

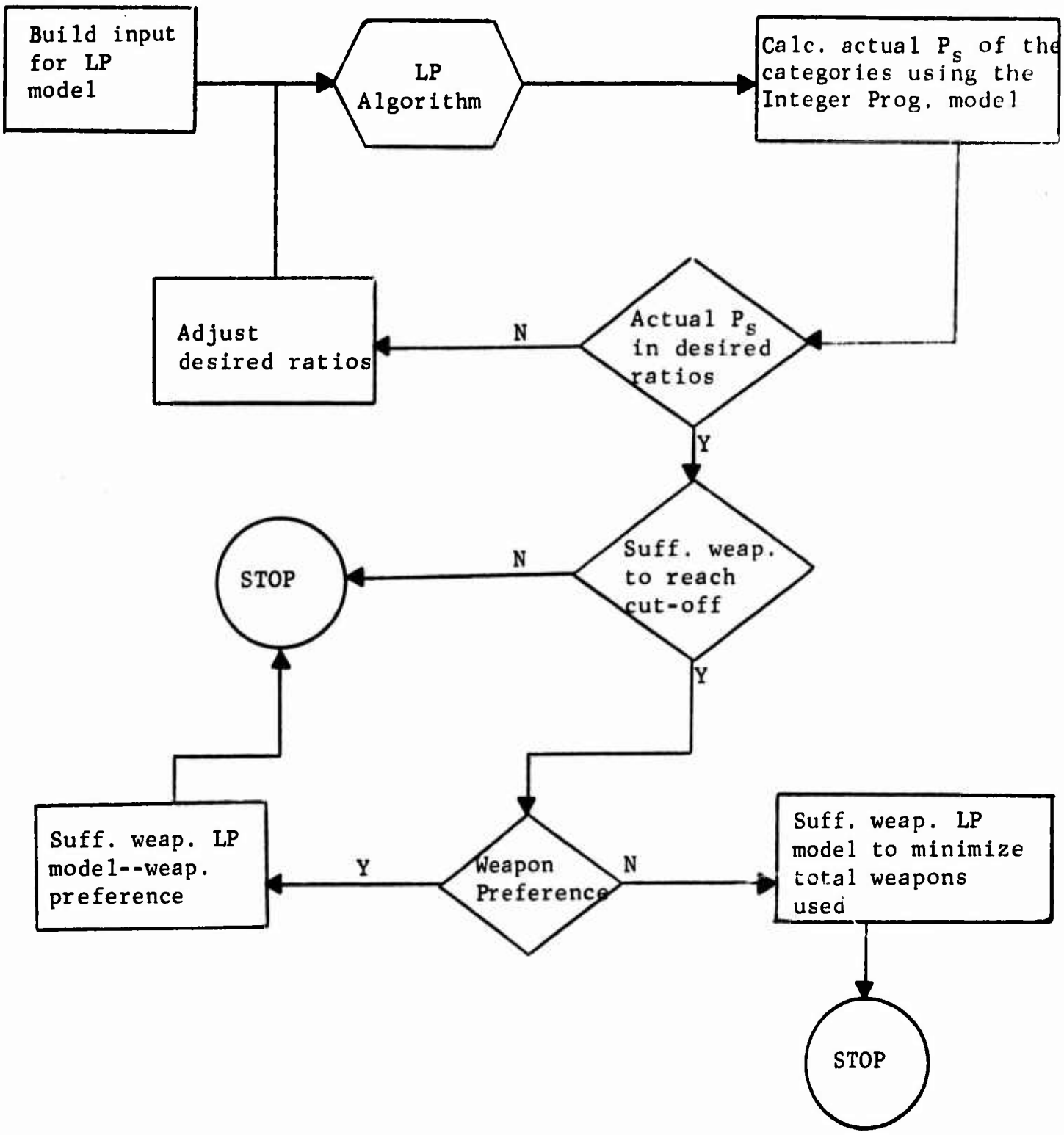

Build input for LP model
LP Algorithm
Calc. actual P_s of the categories using the Integer Prog. model
Adjust desired ratios
N
Actual P_s in desired ratios
Y
STOP
N
Suff. weap. to reach cut-off
Y
Suff. weap. LP model--weap. preference
Y
Weapon Preference
N
Suff. weap. LP model to minimize total weapons used
STOP

FIGURE 5

APPENDIX

MODIFICATION OF THE BASIC LP WEAPON ALLOCATION ALGORITHM--TIME SEQUENCE MODE

The probability of survival of the j^{th} target category is approximately

$$K_j \prod_{i=1}^{m} (S_{ij})^{X_{ij}} \quad , \; j=1,\cdots N$$

Where K_j is the probability of survival of the j^{th} category after the last attack. When targeting in isolation, $K_j=1$, $j=1,\cdots M$

Merely concentrating on Equation 1 of the original algorithm, the problem becomes:

Minimize $\bar{Y}$

Subject to

$$K_j \prod_{i=1}^{m} (S_{ij})^{X_{ij}} \leq C_j \bar{Y} \qquad j=1,\cdots N$$

if both sides are divided by K_j

$$\prod_{i=1}^{m} (S_{ij})^{X_{ij}} \leq \left(\frac{C_j}{K_j}\right) \qquad j=1,\cdots N$$

proceeding just as in the original derivation by taking the logarithms of both sides but treating $\left(\frac{C_j}{K_j}\right)$ as one term

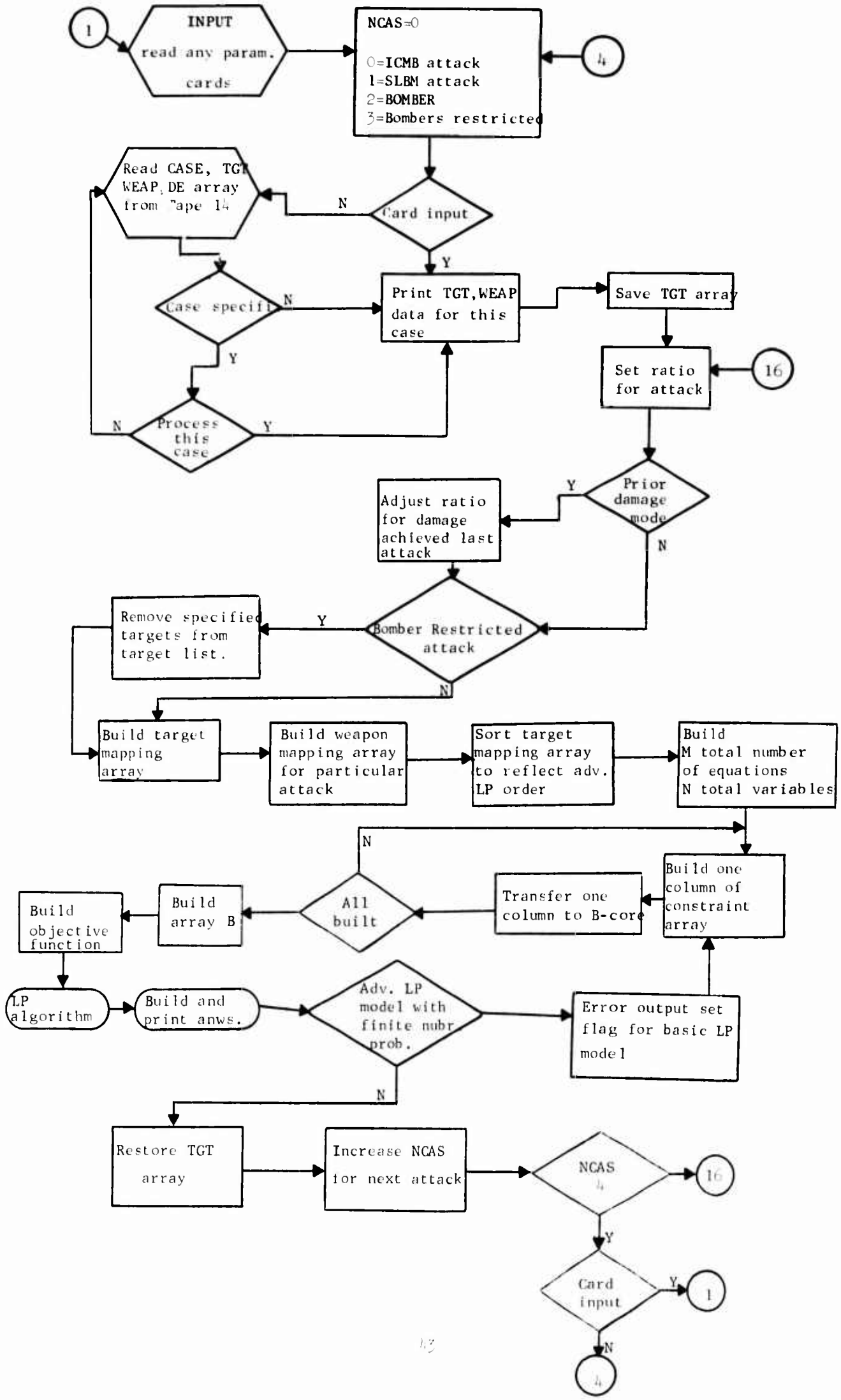
Equation 1 becomes:

$$\sum_{i=1}^{m} (\log S_{ij})\, X_{ij} \leq \log\left(\frac{C_j}{K_j}\right) + \log \bar{Y}$$

or as before

$$\sum_{i=1}^{m} (\log S_{ij})\, X_{ij} - \log \left(\frac{C_j}{K_j} \right) \leq \bar{Y}$$

Thus this set of equations produced will take into account the total damage achieved by the preceeding attacks.


1
INPUT
read any param. cards
NCAS=0
0=ICMB attack
1=SLBM attack
2=BOMBER
3=Bombers restricted
4
Read CASE, TGT WEAP, DE array from Tape 14
N
Card input
Y
Case specif
N
Y
Print TGT,WEAP data for this case
Save TGT array
Set ratio for attack
16
N
Process this case
Y
Y
Prior damage mode
Adjust ratio for damage achieved last attack
N
Remove specified targets from target list.
Y
Bomber Restricted attack
N
Build target mapping array
Build weapon mapping array for particular attack
Sort target mapping array to reflect adv. LP order
Build M total number of equations N total variables
N
Build objective function
Build array B
All built
Transfer one column to B-core
Build one column of constraint array
LP algorithm
Build and print anws.
Adv. LP model with finite nubr prob.
Error output set flag for basic LP model
N
Restore TGT array
Increase NCAS for next attack
NCAS 4
16
Y
Card input
Y
1
N
4

LIST OF REFERENCES

1. Chasen, R.J. "Using Linear Programming as a Simplex Subroutine," Report P-3267. RAND Corporation, Santa Monica, California, November 1965.

2. Pugh, G. E., P.D. Flanagan and R. L. Goodrich. "Analysis of Cross Targeting," Paper 34. Lamda Corporation, Arlington, Virginia, 10 September 1969.

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*

Hq USAFA (Directorate of Faculty Research)
USAF Academy, Colorado 80840

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

A LINEAR PROGRAMMING APPROACH TO WEAPON ALLOCATION

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Research Report

5. AUTHOR(S) *(First name, middle initial, last name)*

Major William T. Hodson III
Major William G. Goodyear
Captain Wolfhart B. Goethert

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
March 1971	60	2

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	RR 71-1
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	None

10. DISTRIBUTION STATEMENT

Distribution of this document is unlimited

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Directorate of Faculty Research (DFSFR) USAF Academy, Colorado 80840

13. ABSTRACT

A two-step optimization procedure using linear programming is employed to obtain a near-optimal solution to the large-scale, multiple-weapon type allocation problem.

Initially, the target system is partitioned into target categories, each of which contains targets of equal worth and similar characteristics. Then, depending upon the requirements of the particular problem addressed, one of three different linear programming models is used to allocate the available supply of weapons among the target categories. Instead of inputing a point value for each target category, as is frequently done in other allocation models, in these models the user indicates a desired ratio of the probability of survival of the various categories.

After the allocation of weapons among the categories has been accomplished, an integer programming model is used to assign weapons to individual targets within each category so as to minimize the average probability of survival of the category.

These models have been programmed for an IBM 7090 computer at Headquarters Strategic Air Command.

DD FORM 1473

Unclassified

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Linear Programming Model Resource Allocation Model Weapon Allocation Model Computerized Models						